VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Síťové systémy pro detekci a prevenci proti narušení

Network intrusion detection and prevention systems

2012

Jan Gomola

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **Jan Gomola**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Síťové systémy pro detekci a prevenci proti narušení
Network Intrusion Detection and Prevention Systems

Zásady pro vypracování:

Pro vyšší zabezpeční síťového provozu je možné využít systémy pro detekci a prevenci proti narušení. Cílem bakalářské práce je popsat, implementovat a ověřit systém IPS v laboratorních podmínkách.
1. Úvod do problematiky a analýza současného stavu.
2. Seznámení s IDS/IPS programem SNORT.
3. Implementace a odladění IPS systému.
4. Ověření funkčnosti pomocí penetračních nástrojů.

Seznam doporučené odborné literatury:

Rehman, R. *Intrusion Detection Systems with Snort*, Pearson Education 2003, ISBN 0-13-1400733-3
Endorf, C., Schultz, E., Mellander, J. *Detekce a prevence počítačového útoku*, Grada 2005, ISBN 80-247-1035-8
Dále podle pokynů vedoucího bakalářské práce.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Pavel Nevlud**

Datum zadání: 19.11.2010
Datum odevzdání: 04.05.2012

VYSOKÁ ŠKOLA BÁŇSKÁ - TECHNICKÁ UNIVERZITA * Fakulta elektrotechniky a informatiky * OSTRAVA *

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlášení studenta

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

Dne: 2. 5. 2012

.....

Podpis

Poděkování

Tímto bych chtěl poděkovat vedoucímu bakalářské práce Ing. Pavlu Nevludovi za odborné vedení, kritické připomínky a rady udělované v rámci konzultací.

Abstrakt

Tato bakalářská práce pojednává o síťových systémech pro detekci a prevenci proti narušení, které slouží pro vyšší zabezpečení síťového provozu. V úvodu práce jsou oba systémy objasněny, stejně jako jejich druhy a způsoby, jakými mohou systémy narušení detekovat. V této práci představuje tyto systémy program Snort. Je schopen tedy narušení detekovat nebo i na něj aktivně reagovat. Dále jsou popsány komponenty Snortu, jeho režimy a pravidla. Jedním z režimů je i režim Inline, který společně s programem Iptables tvoří systém pro prevenci proti narušení. Právě tato kombinace systémů umožňuje aktivně reagovat na narušení. Další částí textu je návrh testovacího prostředí a implementace systému proti narušení. Práce je zakončená testováním implementovaného systému pomocí penetračních nástrojů.

Klíčová slova

IDS, IPS, Snort, Iptables, režim Inline

Abstract

This bachelor thesis deals about network intrusion detection and prevention systems, which are used for strong security of network traffic. Both systems are described in the introduction, as well as their types and ways, how they are able to detect intrusion. Both these systems represents program called the Snort. Snort can detect intrusion and also actively react on intrusion. Further there are described components of Snort, his running modes and rules. One of the modes is Inline mode which in combination with the program Iptables creates intrusion prevention system. This combination is real tool which allows us actively react on intrusion. Another parts of the text are design of test environment and implementation of intrusion prevention system. The thesis is ended with tests of implemented system by penetration tools.

Key words

IDS, IPS, Snort, Iptables, Inline mode

Seznam použitých symbolů a zkratek

ARP (Address Resolution Protocol) – protokol překládající IP adresu na MAC adresu

DoS (Denial of Service) – odepření služby

GRE (Generic Routing Encapsulation) – zapouzdřuje pakety jednoho protokolu do druhého protokolu

HIDS (Host – based Intrusion Detection System) – uzlově orientovaný systém pro detekci proti narušení

HIPS (Host – based Intrusion Prevention System) – uzlově orientovaný systém pro prevenci proti narušení

ICMP (Internet Control Message Protocol) – protokol pro přenos chybových zpráv

IDS (Intrusion Detection System) – systém pro detekci proti narušení

IGRP (Interior Gateway Routing Protocol) – směrovací protokol pro vnitřní brány

IP (Internet Protocol) – protokol pro přenos dat

IPS (Intrusion Prevention System) – systém pro prevenci proti narušení

IPX (Internetwork Packet Exchange) – protokol zaručující předávání nezávislých paketů v interní síti

NIDS (Network – based Intrusion Detection System) – síťově orientovaný systém pro detekci proti narušení

NIPS (Network – based Intrusion Prevention System) – síťově orientovaný systém pro prevenci proti narušení

OSPF (Open Shortest Path First) – směrovací protokol pro větší sítě

RFC (Request for Comments) – žádost o komentáře

RIP (Routing Information Protocol) – směrovací protokol pro menší sítě

RPC (Remote Procedure Call) – vzdálené volání procedur

TCL (Tool Command Language) – skriptovací jazyk

TCP (Transmission Control Protocol) – spojově orientovaný protokol

TOS (Type of Service) – typ služby

TTL (Time to Live) – doba života

UDP (User Datagram Protocol) – nespojově orientovaný protokol

URI (Uniform Resource Identifier) – jednotný identifikátor zdroje

URL (Uniform Resource Locator) – jednotný lokátor zdroje

Obsah

1 Úvod

V dnešní době stále roste počet útoků na aplikace jak z vnější sítě (Internet), tak i z vnitřních sítí (např. Intranet). Je třeba se proti těmto útokům chránit a to pomocí firewallu, který může mít jak hardwarovou, tak i softwarovou podobu. Firewall sice poskytuje ochranu před útoky na služby, ale jen na takové, které nejsou zrovna provozovány a mají tedy příslušné porty zavřeny. Existují však i služby dostupné z venku jako jsou například web, pošta apod. Útočník se snaží využívat chyb v aplikaci nebo přímo v operačním systému serveru, který provozuje danou službu. Proto je potřeba implementovat řešení, které tyto útoky zastaví.

Aby byla ochrana před útoky účinná, je nutné použít další vrstvu zabezpečení, která je schopna monitorovat provoz za otevřenými porty firewallu. Touto vrstvou může být systém IDS. Ještě lepší by však byl systém, který by provoz nejen monitoroval, ale i sám na případný útok reagoval. Takovým systémem je IPS. Pomocí těchto systému detekce a prevence proti narušení lze vytvořit jakousi druhou obrannou linii za firewally určenou k ochraně aplikací komunikujících uvnitř chráněné sítě.

Tato bakalářská práce je členěna celkem do sedmi kapitol. Po úvodní kapitole přecházím k druhé kapitole, kde objasním systémy detekce a prevence proti narušení a následně uvedu jejich srovnání. Další obsáhlá kapitola nese název Snort. Zde popisuji jednotlivé režimy, komponenty a pravidla Snortu. Ve čtvrté, kratší, kapitole popisuji program Iptables, který je neodmyslitelnou částí při spuštění Snortu v Inline režimu. V páté kapitole se zabývám návrhem testovacího prostředí a samotnou implementací IPS systému. V šesté kapitole se věnuji penetračnímu testování, které ověří implementovaný IPS systém. Poslední kapitolou je závěr, ve kterém shrnuji dosažené výsledky.

1.1 Analýza současného stavu

IPS systémy byly vyvinuty koncem 90. let minulého století. Sloužily k řešení nejasností z oblasti pasivního monitorování sítě. První IPS by se z dnešního hlediska daly považovat za IDS, které navíc mohou vydávat preventivní příkazy firewallu.

Postupem času se IPS systémy stále zdokonalovaly. V současné době můžeme považovat IPS za technologii, která dále posouvá schopnosti firewallu směrem dopředu. To znamená, že IPS může rozhodovat o řízení přístupu na základě obsahu datového toku a to nejen podle IP adresy a portu, jak je tomu u tradičního firewallu. Většina IPS využívá ve svých signaturách i cílový port, což vede ke zvýšení výkonnosti a přesnosti monitorování.

Někdo by si mohl myslet, že příchod IPS pomalu vytlačuje systém IDS a zcela ho tak v budoucnu nahradí. Určitě tomu tak není. Abychom dosáhli velmi nízkého počtu falešných poplachů,

musí být IPS stále velmi dobrý IDS. Systémy IPS jsou pak schopny bránit síť ještě před neobjevenými útoky (např. přetečení zásobníku).

Dnešní IPS můžeme nasadit ve dvou variantách:

- Samostatné IPS zařízení – jeho výhoda spočívá ve specializovaném hardwaru k prevenci narušení sítě.
- Integrovaná síťová aplikace – nabízí prevenci narušení napříč celou bezpečnostní infrastrukturou. Typicky se jedná o firewally.

Je určitě finančně a organizačně jednodušší přidat funkcionalitu IPS do firewallu, který je součástí každé moderní podnikové sítě, než nákup a instalace dalších, samostatných zařízení. Troufnu si říct, že v následujících letech se zvýší počet integrovaných IPS. Nicméně na druhou stranu samostatné IPS zařízení ze světa jen tak nezmizí. Jejich použití je vhodné např. v konkrétní části sítě, kde lokální datový provoz nesmí procházet přes firewall. Dále se pak může hodit v případě, kdy jsou firewally a IPS zařízení spravovány různými odděleními zabezpečení sítě.

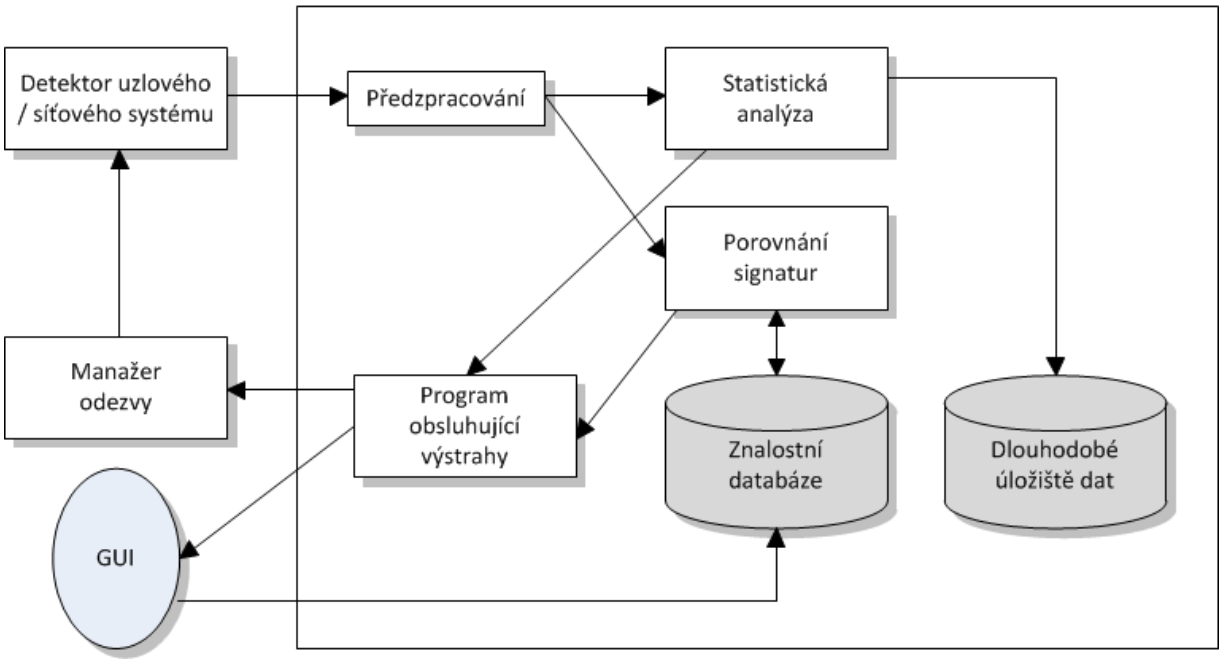
Jaký z těchto dvou způsobů vybereme, záleží na konkrétní situaci v dané organizaci. Tak či onak, jedná se o velmi mocný nástroj pro vyšší zabezpečení sítě, který se bude neustále zlepšovat.

2 Systémy detekce a prevence proti narušení

Jak již bylo zmíněno v úvodu, systémy detekce a prevence proti narušení jsou jakousi druhou obrannou linií za firewallem umožňující monitorovat provoz za otevřenými porty firewallu, popř. i na případný útok reagovat, jak je tomu u IPS. V následujících částích této kapitoly vysvětlím tyto systémy podrobněji.

2.1 Systém detekce narušení (IDS)

Na systém IDS můžeme nahlížet jako na alarm proti zlodějům. Stejně tak jako alarm hlídá nejrůznější objekty, tak i systém IDS monitoruje síťovou komunikaci a komunikační porty. Detekuje tedy neoprávněné průniky do počítačové sítě a další různé odchylky, které nejsou v souladu s nadefinovaným chováním. To znamená, že IDS porovnává právě probíhající komunikaci se signaturami uloženými v databázi. Pokud se výsledek tohoto porovnání neshoduje s normálním nadefinovaným chováním, vygeneruje se výstraha.



Obr. 1: Standardní IDS systém [1]

Systém IDS typicky spadá do třech kategorií:

- HIDS – uzlově orientované IDS
- NIDS – síťově orientované IDS
- Hybridní IDS

2.1.1 HIDS (Host – based IDS)

HIDS vyžaduje určitý software, který je umístěn na tomto systému a může skenovat aktivitu všech uzlových zdrojů. Některý skenuje aktivity systémového a událostního logu. Zapíše libovolnou událost do bezpečnostní databáze a prověří, zda se tyto události neshodují se záznamy závadných událostí obsažených ve znalostní databázi. [1]

2.1.2 NIDS (Network – based IDS)

NIDS se obyčejně zařazuje do sítě sériově a analyzuje síťové pakety, z čehož pak usuzuje na napadení. Přijímá všechny pakety ve zvláštním segmentu sítě, včetně přepínaných sítí (kde to není implicitní chování), pomocí jedné z metod jako např. větvení nebo zrcadlení portů. Pečlivě rekonstruuje provozní (bitový) proud a analyzuje v něm přítomnost vzorů závadného chování. Většina systémů NIDS je vybavena schopností zaznamenávat součinnost, hlásit nebo generovat výstrahu ve sporných případech. Navíc tyto NID schopnosti nabízí většina vysoce výkonných routerů. [1]

2.1.3 Hybridní IDS

Hybridní IDS kombinují HIDS s NIDS. Monitorují tedy jak události odehrávající se na uzlovém systému, tak i síťový provoz.

2.2 Systém prevence proti narušení (IPS)

IPS systémy jsou některými označovány jako nadstavba systémů IDS. IPS systém umí také detekovat útoky jako IDS, ale navíc dokáže útoku aktivně bránit. Senzory se u systému IPS umísťují přímo mezi síťová zařízení. Všechen datový tok pak prochází přímo tímto senzorem a na základě pravidel je možné rozhodnout, zda budou pakety zahozeny či nikoliv.

Podobně jako u IDS se také u IPS systému rozlišují dva druhy:

- HIPS – uzlově orientované IPS
- NIPS – síťově orientované IPS

2.2.1 HIPS (Host – based IPS)

HIPS má své senzory a agenty nainstalované přímo na svém hostiteli, obvykle tedy na počítači s konkrétní IP adresou. Systém IPS je pak těsně spojen s jádrem operačního systému, což je výhodné

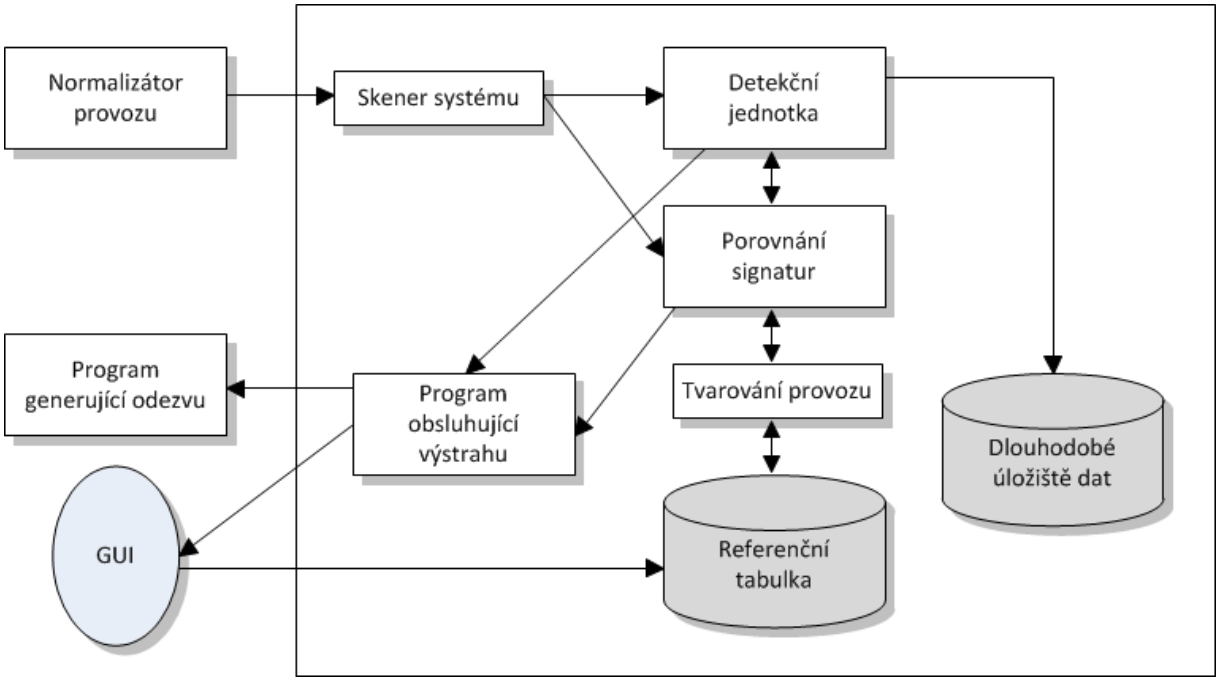
při předcházení útoků a také generování případných výstrah. Aby byl hostitelský počítač chráněn v maximální možné míře, musí být IPS systém spolehlivý, nesmí negativně ovlivňovat výkon, ani blokovat běžnou komunikaci.

2.2.2 NIPS (Network – based IPS)

NIPS je umístěný na síti, kde zachycuje veškerý síťový provoz a kontroluje podezřelé chování. Poskytuje ochranu před všemi útoky typu DoS a to nejen pro klienty a servery, ale i celou síťovou architekturu. Někdy je také nazýván jako Inline IDS.

Typické IPS se skládá ze čtyř částí:

- Normalizátor provozu – přerušuje síťový provoz, provádí rozbor paketů a jejich znovusložení
- Monitor služeb – klasifikuje zpracovávaná data a vytváří referenční tabulku
- Detekční jednotka – porovnává řetězce v paketech s referenční tabulkou a stanovuje, zda bude paket povolen nebo zahozen
- Provozní tvarovací část (tvarovač) – spravuje výsledný datový tok



Obr. 2: Standardní IPS systém [1]

2.3 Způsoby detekce systémů IDS a IPS

Detekce průniků určují, jakým způsobem budou veškerá narušení a napadení detekována. V zásadě se jedná o dva způsoby. Prvním ze způsobů je detekce založená na pravidlech, též někdy zmiňována jako detekce signatury nebo detekce zneužití. Druhým způsobem je pak detekce založená na profilu. Někdy je nazývána také jako detekce anomálií nebo profilově orientovaná detekce.

2.3.1 Detekce založená na pravidlech

Prvním krokem je sběr dat, která se týkají narušení a napadení. Tato data jsou poté vložena do klasifikačního schématu, které obsahuje jméno signatury, jeho identifikátor a popis. Dále pak obsahuje popis možného falešně pozitivního vyhodnocení, informace související s ohrožením a nakonec uživatelskou poznámku, která může nést informaci o konkrétní síti.

Deskriptory vzorů jsou typicky buď založeny na obsahu signatur, které prověřují užitečné zatížení a hlavičku paketu, nebo na kontextu signatur, kdy pak pouze vyhodnocují hlavičku paketu a identifikují výstrahu. Poznamenejme, že deskriptory vzorů mohou být atomické (jednoduché) nebo složené (vícenásobné). Atomický deskriptor při identifikaci výstrah vyžaduje prozkoumání pouze jednoho paketu, zatímco v případě složeného deskriptoru se musí prozkoumat vícenásobný počet paketů. Tyto deskriptory vzorů se pak ukládají do znalostní databáze, která obsahuje kritéria pro analýzu.[1]

Dalším krokem je analýza, která porovnává zformátovaná data a pomocí vzorů je porovnává se znalostní databází. Prohledávají se vzory, které jsou známy jako útoky.

Třetím krokem je odezva. Pokud se vzory shodují, je vygenerována výstraha. Pokud se vzory shodují jen částečně, je prozkoumána následující událost. Tyto částečné shody se analyzují systémem IDS spolu s detektorem stavů, který tak může udržet svůj stav.

V tomto případě vše záleží na aktualizaci signatur. Pokud signatury nejsou aktualizovány, systémy detekce proti narušení ztrácejí svou účinnost. Naštěstí většina těchto systémů umožňuje aktualizaci signatur napadení a to buď manuálně, nebo automaticky.

2.3.2 Detekce založená na profilu

Hlavní myšlenkou detekce založené na profilu je vytvoření modelu normálního chování jednotlivých uživatelů a dalších částí systému, který je vytvořen v určitém časovém intervalu. Za podezřelou událost je pak označena taková událost, která do tohoto modelu nezapadá.

Koncepce založené na anomáliích spadají do tří hlavních kategorií: behaviorální, provozní vzory a protokoly. Behaviorální analýza vyhledává anomálie v těch typech chování, které byly

statisticky znormované, např. vzájemný vztah mezi pakety a tím, co se právě posílá po síti. Protokolová analýza vyhledává narušení síťových protokolů nebo zneužití založené na chování dle RFC. Má přínos v tom, že identifikuje možné útoky, které ještě nejsou veřejně známé nebo mají neznámou signaturu či neexistuje opatření, jak na ně reagovat.[1]

Stejně jako u detekce založené na pravidlech se i zde analytický model skládá ze tří kroků. Prvním krokem je opět sběr dat, která jsou následně uložena v číselném formátu a jsou zformátována. V druhém kroku je provedena analýza. Data, která představují události, jsou pak porovnávána se znalostní databází. Třetím krokem je odezva na dané porovnání. Ta může být spuštěna automaticky, nebo ručně.

Základním rozdílem mezi detekcí založené na profilu oproti ostatním způsobům detekce je, že nedefinují pouze činnosti, které nejsou povolené, ale také ty, kterou jsou povoleny. Výhodou tohoto způsobu je možnost detekování nových, neznámých útoků či narušení, která se jednoduše neshodují s modelem normálního chování. Nevýhodou pak může být např. chování uživatelů, které nelze v žádném případě předvídat, což pak vede k vyvolání falešných útoků.

2.4 Výhody a nevýhody systémů IDS a IPS

Jak jsem již dříve zmiňoval, systémy IPS se velmi podobají svým nastavením systémům IDS. Přesto však každý z těchto systému má své výhody a nevýhody.

Výhody systému IDS jsou následující:

- Může detekovat jak vnější, tak i vnitřní síťově orientované útoky.
- Poskytuje hloubkovou ochranu a dodatečnou vrstvu obrany.
- Poskytuje jednoduchou rozšiřitelnost na celou síť.
- Umožňuje administrátorovi určitě množství napadení.

Následující nevýhody:

- Generuje falešné pozitivní a negativní výsledky.
- Na útok pouze reaguje, nepředchází mu.
- Generuje velké množství dat, která mají být analyzována.
- Vyžaduje personál s velkou odborností, který interpretuje tato data.

Výhody systému IPS spočívají v tom, že:

- Chrání aplikační vrstvu.
- Zabraňuje útoku v reálném čase.
- Poskytuje hloubkovou ochranu.

Nevýhody jsou následující:

- Generuje falešné pozitivní výsledky, a pokud je odezva nastavena na automatickou, způsobuje vážné problémy.
- Vytváří úzká místa v síti.
- Je to nová a nákladná technologie.

3 Snort

Snort je zdrojově otevřený systém pro detekci a prevenci proti narušení. Spojením výhod signatur, protokolu a detekci založené na anomáliích, je Snort nejrozšířenější IDS/IPS technologií na světě. S miliony staženími a téměř čtyřmi sty tisíci registrovaných uživatelů se stal Snort de facto standardem pro IPS. [3]

Princip Snortu spočívá v tom, že hledá vzorky známých útoků a v případě nalezení daného útoku umožňuje provádět různé akce. Probíhající provoz však nepřerušuje. Nejaktuálnější verze Snortu nese označení 2.9.0.5 a běží na mnoha operačních systémech včetně Windows NT/2000/XP, Linuxu a dalších Unixových operačních systémech (Solaris, HP-UX, IRIX, OpenBSD, NetBSD, FreeBSD).

3.1 Režimy Snortu

Snort může běžet v několika různých režimech: režim slídiče (sniffer), režim záznamníku (logger), režim detekce narušení a režim Inline. Právě poslednímu zmíněnému se budu věnovat nejvíce.

3.1.1 Režim slídiče (sniffer mode)

V tomto režimu umožňuje Snort zachytávat data v hlavičce a těle každého paketu, která procházejí sítí a následně je nám zobrazuje na obrazovku. Snort v tomto režimu spustíme příkazem:

```
snort –v
```

Při načtení stránky Googlu (www.google.cz) zachytil Snort celkem 82 paketů. Jeden ze zachycených paketů, který byl zobrazen na obrazovce:

```
03/23-19:50:52.441534 192.168.179.137:58954 -> 209.85.148.99:80
TCP TTL:64 TOS:0x0 ID:52264 IpLen:20 DgmLen:60 DF
******S* Seq: 0x3AE29BDD  Ack: 0x0  Win: 0x3908  TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 4413429 0 NOP WS: 5
```

Pokud bychom chtěli zobrazit hlavičky protokolů IP, TCP, UDP a ICMP, zadáme příkaz:

```
snort –vd
```

Pokud bychom chtěli navíc zobrazit i další popis a hlavičku linkové vrstvy, zadáme příkaz:

```
snort –vde
```

3.1.2 Režim záznámníku

Tento režim je v podstatě shodný s režimem slídiče, avšak jediným rozdílem je, že se paketová data nebo hlavičky zaznamenávají do logu na pevném disku. Pokud tedy chceme spustit Snort v tomto režimu a výstup do adresáře log, zadáme příkaz:

```
snort –v –l /var/log
```

Pro test jsem vyzkoušel příkaz ping na počítač se spuštěným Snortem v tomto režimu. Zachyceno bylo celkem osm paketů a na ukázku uvádím dva (ECHO a ECHO REPLY):

```
03/23-20:12:34.071243 192.168.179.1 -> 192.168.179.137
ICMP TTL:128 TOS:0x0 ID:735 IpLen:20 DgmLen:60
Type:8  Code:0  ID:1   Seq:37  ECHO

03/23-20:12:34.084629 192.168.179.137 -> 192.168.179.1
ICMP TTL:64 TOS:0x0 ID:11579 IpLen:20 DgmLen:60
Type:0  Code:0  ID:1  Seq:37  ECHO REPLY
```

Pokud chceme zadat požadovaný rozsah IP adres sítě, provedeme to příkazem:

```
snort –v –l /var/log –h 10.1.1.0/24
```

Přepínač `–h` a následující argument zajistí, že Snort zaznamená pakety s cílovými adresami v rozsahu 10.1.1.0 až 10.1.1.255. [1]

3.1.3 Režim síťového detektoru narušení

Snort zde odchytává data a následně provádí analýzu v kontextu s předem nadefinovanými pravidly a provádí akce podle nálezu. V tomto režimu se chová jako NIDS a můžeme jej spustit následujícím příkazem:

```
snort -v -l /var/log –h 192.168.179.0/24 –c /snort/etc/snort.conf
```

3.1.4 Režim Inline

Snort se v tomto režimu chová jako IPS. Získává pakety z Iptables a podle předem definovaných pravidel rozhoduje Snort o tom, zda tyto pakety zahodí nebo povolí. Snortu v Inline režimu se věnuji v praktické části, kde popisuji jednotlivé kroky a také uvádím konfiguraci Iptables, která je důležitá pro běh Snortu v tomto režimu. Proto zde uvedu pouze příklad samotného spuštění Snortu v Inline režimu:

```
snort -Q -l /var/log -c /snort/etc/snort.conf
```

3.2 Komponenty Snortu

Snort se skládá z hlavních komponent:

- Jednotka paketového záchytu
- Zásuvné moduly preprocesoru
- Detekční jednotka
- Systém logování a výstrah
- Zásuvné moduly pro výstupy

3.2.1 Jednotka paketového záchytu

Jednotka paketového záchytu sbírá pakety z různých síťových rozhraní počítače pomocí knihoven libpcap nebo WinPcap. Tyto knihovny umožňují aplikacím přijímat datagramy, pomocí nichž jsou přenášena data linkové vrstvy. Síťová karta tento přenos dat zachytává a následně jej předává ovladači rozhraní jádra operačního systému. Když jádro tato data zpracuje, knihovna (libpcap) od něj převezme data a předá je aplikacím Snortu.

3.2.2 Zásuvné moduly preprocesoru

Zásuvné moduly preprocesoru testují a prohlížejí pakety, které byly přijaty z knihoven. Dále pak určují, co se má provést s každým paketem. Jestli se má daný paket analyzovat, změnit, odmítnou či vygenerovat výstrahu pro jeho nebezpečný obsah. Tyto zásuvné moduly preprocesoru jsou velice užitečné, protože nám řeknou, jak zacházet s pakety ještě předtím, než budou poslány do další komponenty.

Preprocesory modifikují URI a URL, aby odpovídaly standardnímu formátu, provádějí stavovou analýzu provozu TCP/IP, detekují skenery portů, dekódují pakety RPC a Telnet, rovněž plní ostatní funkce. Také zmírňují nutnost zacházet s širokým rozsahem nežádoucích a potenciálně zhoubných paketových dat, včetně dat, která by mohla Snort poškodit nebo radikálně zhoršit jeho výkon. Pokud zásuvné moduly preprocesoru daný vstup neodmítnou, předají jej následující komponentě. [1]

3.2.3 Detekční jednotka

Detekční jednotka je nejdůležitější částí Snortu. Systematicky porovnává data uvnitř každého paketu, který přijala. Následně pak provádí základní test libovolné části paketu a kontroluje, zda tento paket obsahuje zvláštní řetězec nebo nějakou hodnotu, která by byla obsažena v daném pravidle

Snortu. Poté spustí další test, který otestuje následující pravidla a to dělá do té doby, než jsou otestována všechna pravidla. Snort má přehled o tom, co vše bylo vykonáno. Každé nalezení jakékoliv shody je pak „zásah“.

3.2.4 Systém logování a výstrah

U této jednotky záleží na tom, co detekční jednotka najde uvnitř paketu. Na základě toho se buď informace zaznamenají do logu, nebo je vytvořena výstraha dle daných pravidel. Log je jednoduchý soubor uložený na disku v adresáři /var/log/snort.

3.2.5 Zásuvné moduly pro výstupy

Hlavním cílem této komponenty je vytvořit informaci, která bude zobrazena v analýze detekce narušení. Snort vytváří výstrahy v závislosti na pravidlech preprocesorů, dekódovacích a detekčních jednotek.

3.3 Pravidla Snortu

Pravidla jsou velice důležitá pro samotný běh Snortu, zvláště pak pokud spustíme Snort jako systém IPS. Snort pak na jejich základě rozhoduje o tom, co se s daným paketem provede. V této části popíši, jak vypadá obecný tvar pravidla. Dále pak oficiální pravidla Snortu, zmíním i neoficiální projekty, které se také věnují pravidlům a nakonec vytvořím i svá vlastní pravidla.

Ukázka obecného tvaru pravidla:

```
akce protokol zdroj_ip zdroj_port směr cíl_ip cíl_port (options)
```

Zdrojová i cílová IP adresa a porty příchozího paketu se porovnávají s výchozími nebo námi vytvořenými pravidly. Jestliže nějaké pravidlo danému paketu odpovídá, pak se vyhodnotí na základě volby pravidla (options). Pokud všechna porovnání souhlasí, vykoná se příslušná akce. Jak můžeme u obecného tvaru pravidla vidět, skládá se ze dvou částí a to hlavičky a volby pravidla.

3.3.1 Hlavička pravidla

Akce

První částí hlavičky pravidla (akce) nám dává možnost určit, o jakou akci se bude jednat. Ve Snortu existuje pět výchozích akcí a to: alert, log, pass, activate a dynamic. Pokud spustíme Snort v inline režimu, přibydou nám k výchozím akcím ještě tři další: drop, reject a sdrop.

Řekněme si tedy, co jednotlivé akce znamenají:

- Alert – vygeneruje výstrahu a následně daný paket zaznamená.
- Log – zaznamená paket.

- Pass – ignoruje paket.
- Active – vygeneruje výstrahu a zavolá další pravidlo.
- Dynamic – tato akce je nečinná, dokud není aktivováno pravidlo „Active“, pak se chová jako „Log“.
- Drop – přidá do Iptables pravidlo, které zahodí a zaznamená daný paket.
- Reject – přidá do Iptables pravidlo, které zahodí a zaznamená daný paket. Pokud je protokolem TCP, pošle TCP reset. Pokud je protokolem UDP, pošle zprávu, že je port nedostupný u ICMP.
- Sdrop – přidá do Iptables pravidlo, které zahodí daný paket, ale nezaznamená jej.

Protokol

Druhou částí hlavičky pravidla je protokol. K dispozici jsou čtyři protokoly, které Snort v současné době analyzuje pro podezřelé chování – TCP, UDP, ICMP a IP. V budoucnu jich však může podporovat více, např. ARP, IGRP, GRE, OSPF, RIP, IPX a další.[4]

IP adresa

Další částí hlavičky je IP adresa. Zdrojovou IP adresou určíme, z jaké adresy budou pakety odcházet. Kam budou dané pakety směrovat, nám určuje cílová IP adresa. Nemusí se však jednat pouze jen o jednu IP adresu, ale můžeme volit celý rozsah IP adres, např. 192.168.1.0/24. Další možností je určení všech IP adres pomocí klíčového slova any. Pokud chceme zvolit všechny IP adresy kromě jediné, využijeme negace, např. !192.168.1.0.

Číslo portu

Podobně jako u IP adresy je tomu tak i u čísla portu. Opět určujeme zdrojový port, ze kterého paket odchází, a cílový port, na který paket směřuje. Čísla portů můžeme zadávat několika způsoby a to zadáním buď jediného čísla portu, nebo celého rozsahu čísel portů. Rozsah určujeme pomocí operátoru :, např. 20:111. Pomocí tohoto operátoru můžeme také určit rozsah čísel portů, který je např. menší nebo roven portu 80 (:80) nebo naopak větší nebo roven portu 80 (80:). Pro určení všech čísel portů použijeme opět klíčové slovo any.

Operátor určující směr

Existují dva druhy operátorů, které určují směr paketu. Prvním z nich je operátor ->. IP adresa a číslo portu před tímto operátorem jsou považovány za zdroj a za operátorem jsou považovány za cíl. Provoz je tedy směřován ze zdroje do cíle. Druhý operátor <> určuje obousměrný provoz. Je to užitečné pro analýzu konverzace obou stran, jako je např. Telnet nebo POP3.

3.3.2 Volba pravidla

Jednotlivé volby jsou v pravidle odděleny středníkem. Klíčová slova jsou od jejich argumentů oddělena dvojtečkou. Volby se v pravidle dělí do čtyř základních kategorií. Jsou to obecná pravidla, pravidla testující data, hlavičku a nakonec tzv. pravidla spuštěná po detekci.

Obecné volby pravidla

Tyto volby v pravidle mají pouze informativní charakter. Poskytují informace o pravidle, ale nemají žádný vliv na rozhodování během detekce. Jsou to např.: msg (textová informace pravidla), gid (slouží k identifikaci, která část Snortu vyvolala výstrahu), sid (ID pravidla Snortu), priority (nastavení priority pravidla), metadata (umožňuje přidat dodatečné informace o pravidle).

Volby pravidla testující data

Tyto volby testují všechna data obsažená v paketu a mohou být vzájemně provázaná.[4] Jsou to např.: content (zkontroluje paket, zda neobsahuje daný řetězec), nocase (nerozlišuje malá a velká písmena u volby content), offset (udává, o kolik bajtů se posune počáteční pozice pro hledání řetězce u volby content).

Volby pravidla testující hlavičku paketu

Tyto volby kontrolují dané hodnoty v hlavičce paketu. Patří mezi ně např.: ttl (kontroluje hodnotu životnosti paketu), dsize (udává maximální velikost paketu, který může být přijat), sameip (kontroluje, zda je zdrojová ip adresa totožná s cílovou ip adresou), flag (používá se pro zjištění, jaký bit příznaku je nastaven v hlavičce paketu [2]). Následující tabulka ukazuje, jaké bity mohou být kontrolovány.

Flag	Argument character used in Snort rules
FIN or Finish Flag	F
SYN or Sync Flag	S
RST or Reset Flag	R
PSH or Push Flag	P
ACK or Acknowledge Flag	A
URG or Urgent Flag	U
Reserved Bit 1	1
Reserved Bit 2	2
No Flag set	0

Tabulka 1 [2]

Volby pravidla spouštěná po detekci

Jedná se o volby pravidla, která jsou spuštěna po samotné detekci. Mezi tyto volby patří např.: logto (všechny pakety, které vyhovují pravidlu, jsou zapsány do samostatného log souboru), react (umožňuje aktivní odpověď, která pošle webovou stránku nebo jiný formát stejného obsahu klientovi a uzavře spojení), replace (u volby content nahradí původní textový obsah novým obsahem, přičemž oba textové řetězce musí mít stejnou délku).

3.3.3 Oficiální pravidla Snortu

Z oficiálních stránek Snortu si můžeme stáhnout celý balík pravidel, který pokrývá všechny známé útoky. Tato pravidla jsou rozdělena do jednotlivých souborů pojmenovaných podle útoku. Např. se jedná o dos.rules, scan.rules, icmp.rules, telnet.rules. V konfiguračním souboru Snortu pak mohu určit, který z těchto souborů obsahující pravidla bude načten při spuštění Snortu. Po stažení jsou všechna pravidla v daném souboru uvedena v komentáři. Pro jejich využití je nutné tento komentář odstranit.

V zásadě máme dvě možnosti, jak získat oficiální pravidla. První možností je stažení nejaktuálnější sady pravidel ihned po jejím vydání, avšak jsme nuceni za tato pravidla uhradit poplatek, resp. si je předplatit. Druhá možnost spočívá ve stažení pravidel s třicetidenním zpožděním od jejich vydání. Podmínkou je pouhá registrace na oficiálních stránkách Snortu.

3.3.4 Neoficiální pravidla

Alternativou oficiálních pravidel ze stránek Snortu jsou pravidla, která můžeme získat z neoficiálních zdrojů a to zcela zdarma. Jedním z nich je i projekt Bleedingsnort, který se mimoto zabývá i dalšími bezpečnostními projekty. Oproti pravidlům z oficiálních stránek jich zde není tolik, ale pro dobrý základ to stačí. Dalším projektem, který se zabývá tvorbou pravidel pro Snort nebo třeba i Suricatu, je Emerging Threats. Oba tyto projekty podporují sponzoři. Některá z těchto volně dostupných pravidel jsem využil a následně otestoval pomocí penetračních nástrojů.

3.3.5 Vlastní pravidla

Pravidel, která si můžeme stáhnout z oficiálních stránek Snortu, je opravdu hodně. Nemusíme však zůstat jen u nich, můžeme si napsat vlastní pravidla, která budou naprosto vyhovovat potřebám naší sítě. Právě možnost psaní vlastních pravidel je jednou z nejlepších vlastností Snortu. Pro psaní vlastních pravidel slouží soubor local.rules. Je vhodné však vytvořit soubor s jiným názvem, např. my.rules. Pokud bychom v budoucnu aktualizovali všechna pravidla, tedy je přepsali, mohlo by dojít i k přepsání právě našich pravidel a to prázdným souborem, což je nežádoucí.

Zde jsou mnou vytvořená pravidla:

```
drop tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"DoS - Syn
Flooding"; flags:S; flow:to_server; detection_filter: track by_src,
count 10, seconds 1; priority:3; sid:4983728;)
```

Snort pomocí tohoto pravidla zahodí a zaznamená do logu všechny pakety směřující na počítač nebo celou síť, která je definovaná proměnnou $HOME_NET. V paketu vyhledává synchronizační příznaky sloužící k navázání spojení. V pravidle je dále filtr, který zachytává deset a více těchto synchronizačních paketů za vteřinu. Je to kvůli tomu, abych předešel falešným útokům.

```
drop icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"Pravdepodobny
utok Ping of Death"; dsize: > 10000; priority:2; sid:7469275;)
```

Snort zahodí a opět zaznamená do logu všechny pakety, které mají velikost větší než 10000 bajtů. Mohlo by se jedna o útok Ping of Death, který posílá velké pakety vedoucí následně k přetečení zásobníku.

```
drop tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"Bad traffic -
stejna IP adresa"; sameip; priority:2; sid:4856302;)
```

Pokud je zdrojová adresa shodná s cílovou adresou, Snort tento paket zahodí a zaznamená do logu.

```
alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"Obsahuje hledany
retezec"; content:"xxx"; nocase; priority:2; sid:37493602;)
```

Snort vyvolá výstrahu a zaznamená do logu všechny pakety, které obsahují řetězec „xxx“. Klíčové slovo nocase nám říká, že daný řetězec může být kombinací malých a velkých písmen.

4 Iptables

Jak jsem již dříve nastínil, Iptables společně se Snortem tvoří tzv. Inline režim. Iptables je také program ovládaný z příkazové řádky, který nám umožňuje nastavovat firewall, resp. jeho jednotlivá pravidla. Tato pravidla nám pak dávají informaci o tom, co se má s daným paketem udělat. Pakety můžeme buď povolit (ACCEPT) nebo zahodit (DROP). Ve výchozím nastavení Iptables je politika nastavena tak, že povoluje všechny pakety. Je to logické, neboť pokud bychom žádný firewall neměli, všechny pakety by byly zahozeny. Při budování firewallu bychom se měli řídit následujícím pravidlem – co není vysloveně povoleno, je zakázáno. O Iptables se hovoří také jako o základním paketovém filtru.

Každý IP datagram s sebou nese vyjma vlastních užitečných dat také hlavičku, obsahující zejména IP adresu původce i adresáta, zdrojový a cílový port specifikující program, kterému je datagram určen, a další informace popisující komunikaci, ke které datagram náleží. Paketový firewall je pak jakýmsi filtrem, který na základě těchto informací rozhoduje o tom, které pakety mohou být připuštěny až k programům, nebo které naopak smějí opustit počítač. [6]

Iptables pracuje se třemi základními tabulkami a každá z nich obsahuje řetězce (chains):

- **Filter** – pokud nenadefinujeme žádnou tabulku, automaticky se použije filter. Je určená k filtrování procházejících paketů a obsahuje řetězce INPUT, OUTPUT a FORWARD. Pravidla z řetězce INPUT se aplikují na pakety, které letí dovnitř sítě. Naopak je to u řetězce OUTPUT, kde se pravidla aplikují pro odchozí pakety. Řetězec FORWARD použijeme ve chvíli, kdy náš server funguje jako router a přeposílá pakety mezi sítěmi. To, co jde přes FORWARD, neprochází pravidly INPUT ani OUTPUT. [5]

- **Nat** – tabulka používaná pro překlad adres. Prvním řetězcem pravidel, který použijeme pro příchozí pakety, je PREROUTING. Pomocí něj můžeme modifikovat cílovou adresu Destination NAT (DNAT) nebo port paketu. Opakem je POSTROUTING, s ním modifikujeme odchozí spojení Source NAT (SNAT), masquerade. [5] Posledním řetězcem pravidel je OUTPUT, ve kterém se pravidla uplatňují ještě před modifikací odchozích paketů.

- **Mangle** – díky této tabulce můžeme upravovat hlavičky paketů (TTL, TOS), značkování a další. Obsahuje všech pět řetězců pravidel a to INPUT, OUTPUT, FORWARD, PREROUTING a POSTROUTING.

  Příklad řetězce INPUT, který zahodí všechny příchozí pakety:

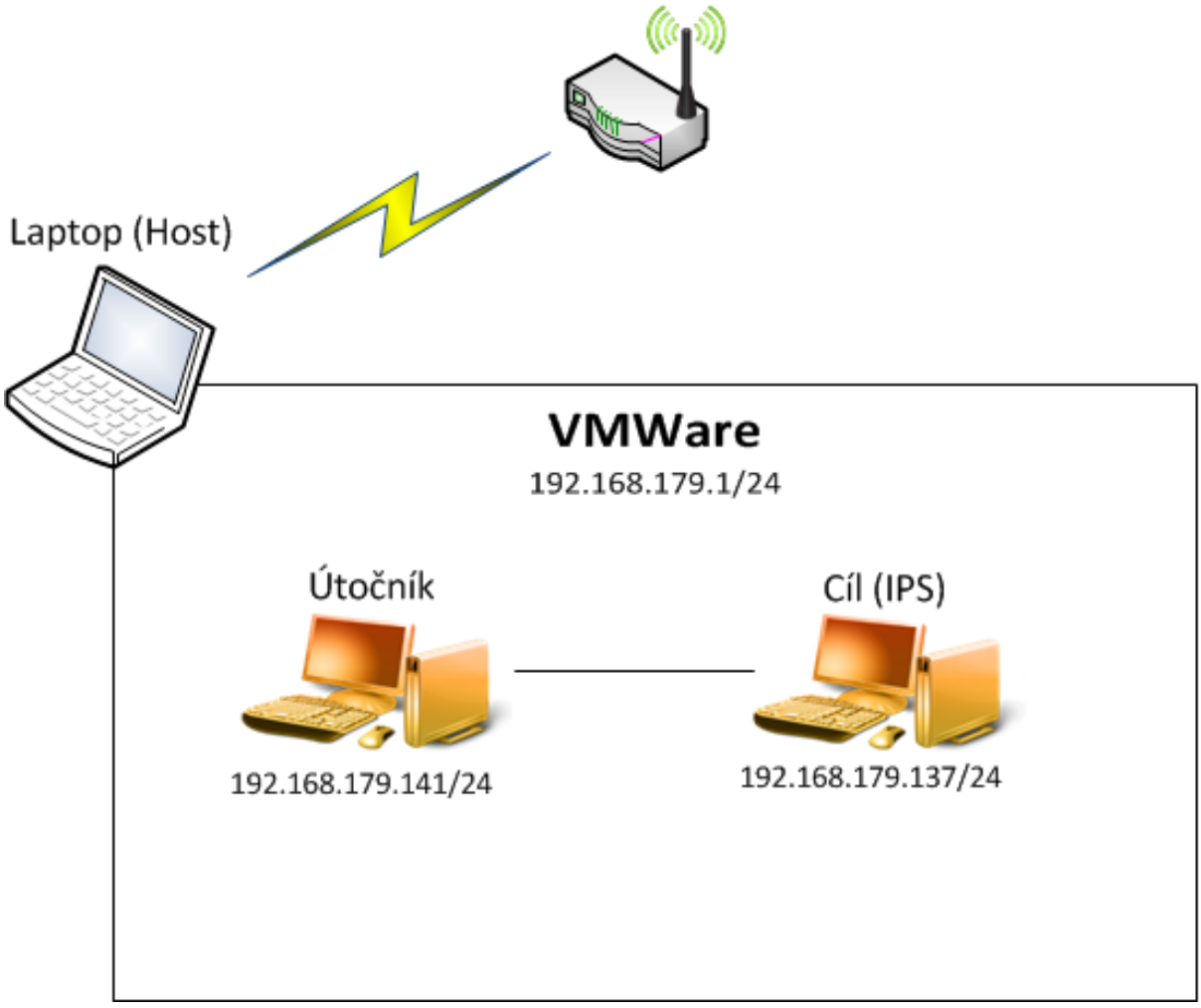
  ```
  iptables -P INPUT DROP
  ```

5 Návrh testovacího prostředí a spuštění IPS systému

V této kapitole popíši návrh mého testovacího prostředí a dále pak spuštění IPS systému, což představuje Snort spuštěný v Inline režimu.

Implementace IPS systému je uvedena v příloze a obsahuje instalaci potřebných balíčků pro Snort, zkompilování a samotnou instalaci Snortu.

5.1 Návrh testovacího prostředí



Obr. 3: Návrh testovacího prostředí

Po dohodě s vedoucím bakalářské práce jsem místo testovacího prostředí v laboratoři zvolil alternativu a to v podobě dvou virtuálních strojů běžících na laptopu. Testovací prostředí se tedy skládá z bezdrátového směrovače a laptopu. Na něm jsou spuštěny dva virtuální stroje a to softwarem VMWare player. První virtuální stroj reprezentuje útočníka. Ten je založen na linuxové distribuci BackTrack 5 R2, která obsahuje spoustu penetračních nástrojů umožňujících testování. V mém případě je testován druhý virtuální stroj,který je založený na linuxové distribuci Ubuntu 11.04. Zde je nainstalován program Snort a je spuštěn v Inline režimu. Virtuální stroj se pak chová jako systém IPS. Vše, co jsem popsal, je uvedeno na obrázku č. 3.

Toto testovací prostředí je naprosto vyhovující pro otestování IPS systému. V tomto případě je zde systém nasazen jako HIPS. V případě NIPS by musel mít virtuální stroj ještě jednu síťovou kartu. Všechen provoz by byl také směřován na systém IPS, za kterým by byl umístěn jeden či několik virtuálních strojů. V praxi by to byla nejspíš celá podniková síť.

5.2 Spuštění Snortu v Inline režimu

Ještě předtím, než spustím Snort, je potřeba přidat do Iptables takové pravidlo, které pošle všechny pakety z jádra do fronty NFQUEUE:

```
iptables -A INPUT -j NFQUEUE
```

Pro ověření, že bylo pravidlo opravdu přidáno do řetězce INPUT, zadám příkaz:

```
iptables -S INPUT
```

Výsledek provedení tohoto příkazu vypadá takto:

```
-P INPUT ACCEPT
-A INPUT -j NFQUEUE --queue-num 0
```

Pokud bych chtěl vidět, jak se změnil řetězec INPUT u tabulky Filter, zadám příkaz:

```
iptables -L INPUT
```

Výstupem jsou pak následující informace:

```
Chain INPUT (policy ACCEPT)
target     prot opt source       destination
NFQUEUE    all  --  anywhere     anywhere   NFQUEUE num 0
```

Nyní mohu spustit program Snort následujícím příkazem:

```
snort -Q --daq nfq --daq-var queue=0 --daq-var device=eth0 --daq-
dir=/usr/lib/daq -l /var/log/snort -c /source/snort-
2.9.0.5/etc/snort.conf
```

Vysvětlení parametrů použitých u spuštění Snortu:

- `-Q` – spuštění Snortu v inline režimu, tedy jako IPS
- `--daq nfq` – zvolení modulu nfq (NFQUEUE)
- `--daq-var queue=0` – parametr daq udávající, z jaké fronty budou pakety předávány
- `--daq-var device=eth0` – nastavení rozhraní
- `--daq-dir` – nastavení cesty, kde je knihovna daq uložena
- `-l` – nastavení cesty pro logy
- `-c` – načtení konfiguračního souboru Snortu

V tento moment rozhoduje Snort na základě pravidel o tom, jaké pakety budou zahozeny a jaké povoleny. Následně je pak pošle zpět do jádra, kde se akce provede.

6 Penetrační testování

Úkolem penetračních testů je především ověření zabezpečení sítě. Snaží se tedy najít co nejvíce chyb a slabých míst v počítačové síti, ale také i v serverech nebo osobním počítači. Obsahují většinou velké množství nástrojů pro skenování portů, spuštěných služeb a jiných. Testy mohou být prováděny testerem jak z vnější, tak i z vnitřní sítě.

Penetrační testy lze provádět několika způsoby. Prvním ze způsobů je tzv. „Black box testing“. V tomto případě nemá tester žádné informace o testovaném systému či vnitřní infrastruktuře. Dalším způsobem je „White box testing“, kde má tester kompletní znalosti o infrastruktuře, kterou testuje. Těmito znalostmi jsou myšleny diagramy, zdrojové kódy a informace o IP adresování. Posledním způsobem je „Grey box testing“. Tímto způsobem jsou myšleny různé variace předchozích dvou způsobů. Tester může např. simulovat zaměstnance ve vnitřní síti.

Penetračních nástrojů je opravdu mnoho. Mohou se vyskytovat samostatně nebo být součástí nějakého softwaru či frameworku. Výbornou volbou je linuxová distribuce BackTrack, která slouží výhradně pro penetrační testování. Aktuální verzí je BackTrack 5 R2. Ta obsahuje spoustu nástrojů, ale také i větší frameworky, mezi které patří i OpenVAS.

6.1 OpenVAS

OpenVAS je Framework několika služeb a nástrojů nabízející komplexní řešení pro vyhodnocení zranitelnosti systému. [7] Použitím OpenVASu v distribuci BackTrack ušetříme čas strávený instalací. Před samotným spuštěním je nutné nejdříve OpenVAS nakonfigurovat. Konfigurace je uvedena také v příloze.

Pro úplné spuštění OpenVASu musíme spustit jednotlivé části a to OpenVAS Manager, OpenVAS Administrator a Greenbone Security Assistant a to následujícími příkazy:

```
openvasmd -p 9390 -a 127.0.0.1
```

```
openvasad -a 127.0.0.1 -p 9393
```

```
gsad --http-only --listen=127.0.0.1 -p 9392
```

Nyní máme možnost si vybrat ze dvou uživatelských rozhraní. Prvním z nich je Greenbone Security Assistant, který spustím příkazem:

```
gsd
```

Druhé rozhraní je v podobě webového prohlížeče, do kterého, v mém případě, napíši IP adresu v kombinaci s daným portem:

```
http://127.0.0.1:9392
```

Zde se přihlásím pod administrátorským účtem. Po přihlášení mohu vybírat testy a volit konkrétní cíle v podobě IP adresy. V celku je na výběr ze čtyř testů, které jsou rozděleny podle náročnosti. Od toho se také odvíjí čas celkového testu. Jsou to „Full and fast“ a „Full and fast ultimate“, které patří mezi rychlejší testy. Dále to jsou pak „Full and very deep“ a „Full and very deep ultimate“, které jsou o něco pomalejší, ale zato testují systém důkladněji.

Pro otestování IPS systému jsem zvolil test „Full and very deep ultimate“, který trval zhruba 25 minut. Snort na základě tohoto testu vygeneroval log soubor, který je uveden v příloze.

6.2 Hping

Hping je nástroj pro generování a analyzování paketů TCP/IP protokolu. [8] Používá se k pokročilému skenování portů, testování firewallu, ale i k měření výkonu sítě. Umí posílat i TCP nebo UDP pakety a umožňuje uživateli upravovat jejich hlavičky, což klasický ping neumí. Nová verze nese označení Hping3 a používá skriptovací jazyk TCL. [8] Pomocí něj můžeme generovat velké množství různých druhů útoků. Spuštění Hpingu je v příkazové řádce velice jednoduché. V podstatě se jedná o zadávání parametrů, kterých je v případě Hpingu opravdu mnoho.

Útoky, které jsem provedl Hpingem:

```
hping3 -S 192.168.179.137 -p 80 --flood
```

Jedná se o útok typu DoS, konkrétně Syn Flooding. To znamená, že útočník posílá mnoho paketů cílovému počítači s příznakem SYN, ale dále již neodpovídá. Právě parametr `-S` značí posílání příznaku SYN. Jelikož je ve výchozím nastavení určen protokol TCP, nemusíme ho do příkazu psát. Cíl směřuje na danou IP adresu na port 80 (parametr `-p`). Parametr `--flood` nám říká, že posílá pakety na cíl co nejrychleji. Útok jsem spustil na pět vteřin a Snort zaznamenal do logu 13890 paketů.

```
hping3 -S -i u1000 -a 84.84.84.84 192.168.179.137 -p 80
```

Další zajímavostí je podvrhnutí zdrojové IP adresy. V tomto případě jsem podvrhnul původní IP adresu útočníka za IP adresu 84.84.84.84 a to parametrem `-a`. Také je možno použít parametr `--rand-source`, který každý paket pošle z jiné IP adresy. Další parametrem je `-i`, který určuje interval posílání paketů. V případě tohoto útoku je posíláno 100 paketů za vteřinu. Tento útok trvající pět vteřin zachytí stejné pravidlo jako v předchozím případě a Snortem je do logu zaznamenáno 4465 paketů.

Pomocí Hpingu jsem také zkusil otestovat pravidla dekodéru a preprocesoru, která jsem povolil při kompilaci Snortu. Jak jsem již zmiňoval, slouží pro zachycení abnormálního provozu. Při spuštění Snortu jako IPS se zachycení abnormálního provozu nikde nevypíše ani neuloží, jen tento provoz jednoduše blokuje. Při spuštění Snortu jako IDS jsou na obrazovku vypisovány varování o

zachyceném paketu a to právě díky pravidlům dekodéru a preprocesoru. Vyzkoušel jsem následující příkazy:

```
hping3 -1 -a 192.168.179.137 192.168.179.137
```

Klasický ICMP ping, ve kterém jsem podvrhnul zdrojovou IP adresu za cílovou. Snort tento příkaz vyhodnotí jako špatný provoz a vypíše na obrazovku toto varování:

```
(snort decoder) WARNING: Bad Traffic Same Src/Dst IP
04/23-19:41:17.073936 192.168.179.137 -> 192.168.179.137
ICMP TTL:64 TOS:0x0 ID:56679 IpLen:20 DgmLen:28
Type:8  Code:0  ID:531   Seq:0  ECHO
```

```
hping3 -1 -d 65468 192.168.179.137
```

ICMP ping, který pošle paket s téměř maximálně možnou velikostí. Jedná se o útok typu DoS, konkrétně Ping of death. Tento útok je velice starý a nové systémy s ním již nemají žádný problém. Na obrazovku je vypsáno následující varování:

```
(snort_decoder) WARNING: IP dgm len > captured len!
04/23-19:40:46.148886 192.168.179.141 -> 192.168.179.137
ICMP TTL:64 TOS:0x0 ID:1 IpLen:20 DgmLen:376
Frag Offset: 0x1FCC   Frag Size: 0x0164
```

6.3 Nmap

Nmap je nástroj sloužící ke skenování jednotlivých počítačů, ale i celých počítačových sítí. Skenuje tedy otevřené porty a identifikuje služby běžící na těchto portech. Dále pak pomocí něj můžeme také zjistit typ operačního systému, který běží na daném počítači. [9] Nmap můžeme spustit na všech běžných operačních systémech a podobně jako Hping se spouští v příkazovém řádku. Je možné také použít grafické uživatelské rozhraní s názvem Zenmap, které slouží pro jednoduché nastavení parametrů a následné spuštění Nmapu.

Zenmap obsahuje několik profilů, které představují jednotlivé testy a to od rychlých a jednoduchých po pomalé a rozsáhlé. Profily lze editovat, tedy přidávat či odebírat parametry, nebo vytvářet nové profily podle našich představ. Pro test jsem zvolil profil „Slow comprehensive scan“, což je pomalý rozsáhlý scan. Příkaz tohoto testu vypadá následovně:

```
nmap -sS -sU -T4 -A -v -PE -PP -PS80,443 -PA3389 -PU40125 -PY -g 53
--script "default or (discovery and safe)" 192.168.179.137
```

Po provedení tohoto testu na cílový virtuální stroj, kde nebyl spuštěn Snort v Inline režimu, našel Nmap otevřené porty 21, 22, 80 a 5353. Poté jsem spustil test ještě jednou na cílový virtuální stroj, kde již byl spuštěn Snort v Inline režimu. Nmap zde našel pouze jeden otevřený port a to 5353. Průběh obou scanů je uveden v příloze stejně jako log soubor vygenerovaný Snortem. Test trval zhruba 24 minut.

```
nmap 192.168.179.137
```

Jedná se o nejjednodušší příkaz Nmapu, který během okamžiku vypíše všechny otevřené porty. V tomto příkazu bez dalších parametrů se testuje prvních tisíc portů. Pokud je Snort spuštěný, postará se o to, aby příkaz žádné porty nevypsal. Pokud není spuštěn, vypíše opět porty 21, 22 a 80. Log soubor je uveden v příloze.

```
nmap -sA -p 0-65535 -T4 -v 192.168.179.137
```

Tento příkaz oskenuje všech 65536 portů a je funkční pouze na UNIXových platformách. Parametr `-sA` označuje ACK scan, který je určen spíše pro detekci pravidel. Ve výpisu tedy žádné otevřené porty nenajdeme. Scan bez spuštěného IPS systému byl hotový do deseti vteřin. Se spuštěným IPS systémem trval zhruba dvacet minut. Log soubor vygenerovaný Snortem je opět v příloze.

7 Závěr

Cílem této bakalářské práce bylo navrhnout, implementovat a ověřit IPS systém pomocí penetračních nástrojů. Implementace IPS systému v podobě Snortu spuštěného v Inline režimu proběhla bez větších problémů. Pro testování IPS systému jsem využil penetrační nástroje, konkrétně OpenVAS, Hping a Nmap.

Abych ještě lépe viděl rozdíly mezi detekcí a prevencí proti narušení, otestoval jsem i systém IDS, což v mém případě znamenalo spuštění Snortu v režimu síťové detekce narušení. Pomocí nástroje Nmap sloužícího ke skenování portů jsem provedl test obou systémů. Při testu systému IDS vypsal Nmap všechny otevřené porty. V případě IPS systému nevypsal žádný otevřený port a to díky aktivní ochraně, kterou umožňuje.

Pro ještě lepší ochranu systému a tedy i zachycení dalších nebezpečných útoků jsem vytvořil svá vlastní pravidla. Tato pravidla jsem pak otestoval pomocí nástroje Hping, kterým jsem i mimo jiné generoval útok DoS, konkrétně TCP Syn flooding.

Z pohledu dalšího vývoje je možné tuto práci rozšířit o detailnější popis jednotlivých preprocesorů Snortu, kterými jsou např. Frag3, Stream5, sfPortscan a mnoho jiných. Dalším bodem rozšiřující práci by mohl být program Pytbull využitý v rámci penetračního testování. Obsahuje více než 300 testů rozdělených do devíti kategorií.

U programu Pytbull není dokumentace příliš rozsáhlá, nicméně rozhodl jsem se jej pro penetrační testování použít. Při testech systému IDS testy proběhly úspěšně, u systému IPS nikoliv. Proto jsem jej také neuvedl v kapitole věnující se penetračnímu testování.

8 Literatura

[1] Schultz Eugene, Mellander Jim, Endorf Carl: *Detekce a prevence počítačového útoku.* Grada, 2005. 356 s. ISBN: 80-247-1035-8

[2] Rafeeq Ur Rehman: *Intrusion Detection Systems with Snort. PrenticeHall*, 2003, 275 s. ISBN: 0-13-140733-3

[3] Martin Roesch: *Snort*

Dostupný z URL: http://www.snort.org/

[4] Martin Roesch: *Snort Users Manual 2.9.2*

Dostupný z URL: http://www.snort.org/assets/166/snort_manual.pdf

[5] Csaba Botoš: *Vše o Iptables*

Dostupný z URL: http://www.root.cz/clanky/vse-o-iptables-uvod/

[6] Miroslav Petříček: *Stavíme firewall (1)*

Dostupný z URL: http://www.root.cz/clanky/stavime-firewall-1/

[7] Tim Brown: *OpenVAS*

Dostupný z URL: http://www.openvas.org/

[8] Salvatore Sanfilippo: *Hping*

Dostupný z URL: http://wiki.hping.org/

[9] Gordon Lyon: *Nmap*

Dostupný z URL: http://nmap.org/

9 Seznam příloh

Složka Konfigurace a instalace – *Konfigurace_a_instalace.pdf*, 7 str., obsahuje části:

- Instalace potřebných balíčků pro Snort
- Zkompilování a instalace Snortu
- Konfigurace OpenVAS

Složka OpenVAS – *alert.txt, snort.log.1334588692*

Složka Hping – obsahuje složky:

- Syn Flooding – *alert.txt, snort.log.1334834587*
- Syn Flooding (Spoofing) – *alert.txt, snort.log.1334838204*

Složka Nmap – obsahuje složky a soubory:

- ACK scan – *alert.txt, snort.log.1335176299*
- Simple scan – *alert.txt, snort.log.1335216548*
- Slow comprehensive scan – *alert.txt, snort.log.1335260519*
- *Vystup_scanu_IDS.txt*
- *Vystup_scanu_IPS.txt*

Složka Výpisy Snortu – obsahuje soubory:

- *Inline.txt*
- *Vypis.txt*